REMARKS/ARGUMENTS

In the Office Action, the Examiner noted that claims 1-25 are pending in the application. The Examiner additionally stated that claims 1-25 are rejected. By this communication, claims 1, 7-8, 16, 20-21, and 25 are amended. Hence, claims 1-25 are pending in the application.

Applicant hereby requests further examination and reconsideration of the application, in view of the foregoing amendments.

In the Specification

Applicant has amended the specification to secure a substantial correspondence between the claims amended herein and the remainder of the specification. No new matter is presented.

In the Claims

Rejections Under 35 U.S.C. §103(a)

The Examiner rejected claims 1-25 under 35 U.S.C. 103(a) as being unpatentable over Kessler, US6789147 (hereinafter, "Kessler"), in view of Colavin, U.S. Publication No. 20040103263 (hereinafter, "Colavin"), and further in view of Miller, US6081884 (hereinafter, "Miller"). Applicant respectfully traverses the Examiner's rejections.

In reply to Applicant's arguments submitted with the amendment of 11/18/2008, the Examiner pointed out that arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references.

Applicant respectfully replies that the arguments submitted were indeed directed toward a combination of the references and points were made in disputation of the Examiner's assertions for individual references. For example, it was argued that:

> "Applicant has thoroughly studied the teachings of Kessler, Colavin, and Miller, both alone and in combination, and finds that <u>Kessler and Colavin</u> fail to teach any form of an x86-compatible microprocessor. As has been previously submitted, <u>Kessler teaches a security co-processor interface.</u>

He does not teach an x86 integer unit, x86 floating point unit, x86 MMX unit, or x86 SSE unit. Kessler fails to teach or suggest the capability to correctly execute a majority of the application programs that are designed to be executed on an x86 microprocessor.

Colavin teaches clustered VLIW processing elements, coupled by a runtime reconfigurable inter-cluster interconnect to form a coprocessor executing only those portions of a program having high instruction level parallelism. (Abstract). Applicant respectfully submits that Colavin clearly states in this brief sentence that his technique is not capable of the execution of application programs, but *only those portions of a program having high instruction level parallelism.* Thus, Applicant respectfully asserts that the Examiner's conclusion Colavin's co-processor executes the actual application program, which meets the limitation of said cryptographic instruction is one of the instructions in an application program, wherein said application is executed by said microprocessor to obtain expected results, is refuted by the above excerpt from Colavin. Furthermore, Colavin does not teach an x86 integer unit, x86 floating point unit, x86 MMX unit, or x86 SSE unit. In addition, Colavin fails to teach or suggest the capability to correctly execute a majority of the application programs that are designed to be executed on an x86 microprocessor.

Regarding the teachings of Miller, Applicant indeed agrees that the x86 instruction set has been widely accepted because of it's compatibility with a large amount of software. However, Miller in no way suggests that a microprocessor be provided that is both x86 compatible and that has an x86 cryptographic instruction that takes advantage of an integral cryptographic unit. Rather, Miller's motivation is to provide an alignment unit within a microprocessor that is configured to detect variable length instructions as they are fetched from an instruction cache and then embed the variable length instruction within a long instruction word, thus

> achieving the benefits of a long instruction word format, while still retaining backward compatibility with variable length instructions. Miller's microprocessor is indeed configured according to some aspects of the x86 architecture, but he fails to teach or suggest a cryptographic instruction for execution, or furthermore, any execution unit that may be capable of performing a cryptographic operation.
>
> Applicant respectfully submits that neither Kessler, Colavin, nor Miller teach an x86-compatible microprocessor that executes cryptographic instruction as part of an application program, and that performs a specified cryptographic operation. In addition, it is respectfully asserted that none of the noted references would point one skilled in the art to provide a cryptographic instruction that can be executed by an x86-compatible microprocessor.
>
> By combining the two references, one skilled in the art would be led to conclude that Kessler's coprocessor may be useful in an x86 environment because it could offload cryptographic functions which would otherwise have to be performed via operating system intensive subroutine calls, and that Colavin's technique may be useful for those portions of a program having high instruction level parallelism."

In the excerpt above, Applicant pointed out that if none of the cited references teach a recited element of the claimed invention (e.g., "an x86-compatible microprocessor"), then it does not follow that the cited references can be combined to yield that limitation. In addition to showing that none of the references recite, suggest, or allude to a recited element, as noted above, Applicant also argued against various combinations of the cited references. Accordingly, it is submitted that Applicant's arguments in the previous amendment were indeed directed toward the combination of references cited, and not individually.

Referring to claims 1 and 21, the Examiner noted that Kessler discloses a co-processor that includes multiple execution units (Figure 2) wherein each of the execution units includes an execution queue to store cryptographic instructions received by the co-processor (Figure 8). The Examiner noted that this meets the limitation of a fetch logic, disposed within a microprocessor, configured to receive a cryptographic instruction as a part of an instruction flow executing on said microprocessor, wherein said cryptographic instruction prescribes one of the cryptographic operations.

The Examiner also stated that the execution units include a plurality of operation blocks that correspond to different cryptographic operations that are used depending upon the type of instruction received in the execution queue (Figure 8 & Col. 9, lines 7-43), which meets the limitation of wherein said cryptographic instruction prescribes one of a plurality of cryptographic algorithms, algorithm logic, operatively coupled to said cryptographic instruction, configured to direct said microprocessor to execute said one of the cryptographic operations according to said one of a plurality of cryptographic algorithms.

The Examiner further observed that using the appropriate operation block, the corresponding cryptographic algorithm is used when processing the received instruction (Col. 9, lines 28-43), which meets the limitation of execution logic, operatively coupled to said algorithm logic, configured to execute said one of the cryptographic operations.

The Examiner also noted that The operation blocks correspond to cryptographic algorithms such as AES, 3DES, DES, and RC4 (Figures 5 & 8), which meets the limitation of executing a plurality of cryptographic rounds required to complete said one of the cryptographic operations.

The Examiner noted that Kessler does not specify that the co-processor executes that program that includes the cryptographic operations, but that Colavin discloses a host and co-processor configuration wherein the co-processor executes the actual application program (Abstract & [0018]), which meets the limitation of said cryptographic instruction is one of the instructions in an application program, wherein said application is executed by said microprocessor to obtain expected results. The Examiner observed that it would have been obvious to one of ordinary skill in the art at the time the invention was made for the co-processor of Kessler to execute the actual application program as described by Colavin in order to efficiently execute programs with high instruction level parallelism as taught by Colavin ([0002]).

The Examiner stated that Kessler does not specify that the co-processor utilizes the x86 instruction set, however, it would have been obvious to one of ordinary skill in the art at the time the invention was made for the co-processor described in Kessler to implement the x86 instruction set because the x86 instruction set has been widely accepted because of it's compatibility with a large amount of software as taught by Miller (Col. 2, lines 9-14). The Examiner added that Applicant's specification shows that integer instructions are inherent to the x86 instruction set (Page 27), and therefore, when implementing the x86 instruction set in the co-processor of Kessler, as previously described, the execution units would effectively operate as a "integer unit" as claimed.

By this communication, Applicant has amended claims 1, 16, and 21 to recite a "single, atomic cryptographic instruction," which, as one skilled in the art will appreciate, operates transparently in the presence of interrupts, as are commonly encountered when a general purpose microprocessor, such as an x86-compatible microprocessor, is executing any one of the application programs that are designed to be executed on an x86 microprocessor, such as Microsoft WindowsXP and Microsoft Outlook, as is taught in the instant specification.

The independent claims also recited "an x86-compatible microprocessor." By way of summary, the specification teaches that in order for any device to be deemed equivalent to an x86-compatible microprocessor as defined above, it must be 1) compatible with the x86 architecture, 2) it must include an x86 integer unit, 3) it must include an x86 floating point unit, 4) it must include an x86 MMX unit, and 5) it must include an x86 SSE unit. In addition, an equivalent device must 6) correctly execute a majority of the application programs that are designed to be executed on an x86 microprocessor.

Applicant has thoroughly studied the teachings of Kessler, Colavin, and Miller, both alone and in combination, and finds that Kessler and Colavin fail to teach any form of an x86-compatible microprocessor. As has been previously submitted, Kessler teaches a security co-processor interface. He does not teach an x86 integer unit, x86 floating point unit, x86 MMX unit, or x86 SSE unit. Kessler fails to teach or suggest the capability to correctly execute a majority of the application programs that are designed to be executed on an x86 microprocessor.

Colavin teaches clustered VLIW processing elements, coupled by a runtime reconfigurable inter-cluster interconnect to form a coprocessor executing only those portions of a program having high instruction level parallelism. (Abstract). Applicant respectfully submits that Colavin clearly states in this brief sentence that his technique is not capable of the execution of application programs, but *only those portions of a program having high instruction level parallelism.* Thus, Applicant respectfully asserts that the Examiner's conclusion Colavin's co-processor executes the actual application program, which meets the limitation of said cryptographic instruction is one of the instructions in an application program, wherein said application is executed by said microprocessor to obtain expected results, is refuted by the above excerpt from Colavin. Furthermore, Colavin does not teach an x86 integer unit, x86 floating point unit, x86 MMX unit, or x86 SSE unit. In addition, Colavin fails to teach or suggest the capability to correctly execute a majority of the application programs that are designed to be executed on an x86 microprocessor.

Regarding the teachings of Miller, Applicant indeed agrees that the x86 instruction set has been widely accepted because of it's compatibility with a large amount of software. However, Miller in no way suggests that a microprocessor be provided that is both x86 compatible and that has an x86 cryptographic instruction that takes advantage of an integral cryptographic unit. Rather, Miller's motivation is to provide an alignment unit within a microprocessor that is configured to detect variable length instructions as they are fetched from an instruction cache and then embed the variable length instruction within a long instruction word, thus achieving the benefits of a long instruction word format, while still retaining backward compatibility with variable length instructions. Miller's microprocessor is indeed configured according to some aspects of the x86 architecture, but he fails to teach or suggest a cryptographic instruction for execution, or furthermore, any execution unit that may be capable of performing a cryptographic operation.

By combining the three references, one skilled in the art would be led to conclude that Kessler's coprocessor may be useful in an x86 environment because it could offload cryptographic functions which would otherwise have to be performed via operating system intensive subroutine calls, and that Colavin's technique may be useful for those portions of a program having high instruction level parallelism.

As one skilled in the art will appreciate, an x86-compatible microprocessor is a general purpose microprocessor that is configured to execution commonly used application programs such as Microsoft WindowsXP and Microsoft Outlook. These two applications are specifically noted in the instant specification. And as one skilled will agree, both of these applications comprise provisions to interrupt the microprocessor as well as program control transfer instructions (e.g., JMP). Yet, Kessler specifically teaches away from employing a general purpose microprocessor, in that he asserts:

> Performing tasks to establish a secure session is processor intensive. If a general purpose microprocessor, acting as a host processor for a network element, performs these tasks, then the network element's system performance will suffer because resources will be consumed for the tasks. (col. 1, lines 56-61)

Applicant thus submits that it is difficult to understand how Kessler, Colavin, and Miller could be practically combined in any fashion. Kessler teaches away from the use of a general purpose microprocessor to perform security operations. If the security operations taught by Kessler were to be performed by a host processor, it would thus defeat his stated purpose, which is to offload tasks from the host processor. (col. 1, line 66 – col. 2, line 39). Furthermore, only the Kessler reference refers to cryptographic operations, and only in the context of a coprocessor interface.

Applicant also fails to appreciate how it would have been obvious to one of ordinary skill in the art at the time the invention was made for the co-processor described in Kessler to implement the x86 instruction set because the x86 instruction set has been widely accepted because of it's compatibility with a large amount of software as taught by Miller. As is pointed out in the instant specification, message encryption and decryption are very common operations, and there has been a noted desire in the community to accelerate these operations because they have heretofore been performed via either dedicated calls to software subroutines, or by co-processors such as that taught by Kessler. And there are numerous manufacturers of x86-compatible microprocessors to include Intel Corporation and Advance Micro Devices. Yet, none of these manufacturers have been able—to date—to develop an x86-compatible microprocessor that meets the elements and limitations recited in claims 1, 16, and 21. Accordingly, it is respectfully asserted that one of ordinary skill in the art at the time of the invention would have not understood how Kessler, Colavin, and Miller could have been combined to yield the elements and limitations of the independent claims.

In addition to the above showings, Applicant respectfully notes that the combination of Kessler, Colavin, and Miller fails to teach or suggest a "single atomic cryptographic instruction" for execution on an "x86-compatible microprocessor." Applicant has searched all of the references to determine if there is any teaching, or practical motivation provided that would even allude to a single atomic cryptographic instruction. There is none. The references utterly fail to teach this limitation.

Based upon the above arguments and instant amendments, Applicant respectfully requests that the rejection of claim 1 be withdrawn.

Claim 21 recites substantially the same limitations as have been argued above as being allowable over the combination of Kessler, Colavin, and Miller. Accordingly, it is requested that the rejection of claim 21 be withdrawn as well

With respect to claims 2-15, these claims depend from claim 1 and add further limitations that are neither anticipated nor made obvious by the combination of Kessler, Colavin, and Miller. Accordingly, Applicant respectfully requests that the Examiner withdraw the rejections of claims 2-15.

With respect to claims 22-25, these claims depend from claim 21 and add further limitations that are neither anticipated nor made obvious by the combination of Kessler, Colavin, and Miller. Accordingly, Applicant respectfully requests that the Examiner withdraw the rejections of claims 22-25.

As per claim 16, the Examiner noted that Kessler discloses a co-processor that includes multiple execution units (Figure 2) wherein each of the execution units includes an execution queue to store cryptographic instructions received by the co-processor (Figure 8), which meets the limitation of a cryptographic unit within a microprocessor, configured to execute one of the cryptographic operations response to receipt of a cryptographic instruction that prescribes said one of the cryptographic operations, wherein said cryptographic instruction is one of the instructions in an application program that are fetched from memory by fetch logic in said microprocessor. The Examiner additionally observed that the execution units include a plurality of operation blocks that correspond to different cryptographic operations that are used depending upon the type of instruction received in the execution queue (Figure 8 & Col. 9, lines 7-43), which meets the limitation of an algorithm field, configured to prescribed one of a plurality of cryptographic algorithms to be employed when executing said one of the cryptographic operations. The Examiner noted that using the appropriate operation block, the corresponding cryptographic algorithm is used when processing the received instruction (Col. 9, lines 28-43), which meets the limitation of algorithm logic, operatively coupled to said cryptography unit, configured to direct said device to perform said one of the cryptographic operations according to said one of the plurality of cryptographic algorithms.

The Examiner also noted that Kessler does not specify that the co-processor executes that program that includes the cryptographic operations, but that Colavin discloses a host and co-processor configuration wherein the co-processor executes the actual application program (Abstract & [0018]), which meets the limitation of wherein said microprocessor executes said application program to obtain expected results. The Examiner then observed that it would have been obvious to one of ordinary skill in the art at the time the invention was made for the co-processor of Kessler to execute the actual application program as described by Colavin in order to efficiently execute programs with high instruction level parallelism as taught by Colavin ([0002]).

The Examiner further noted that Kessler does not specify that the coprocessor utilizes the x86 instruction set, however, it would have been obvious to one of ordinary skill in the art at the time the invention was made for the co-processor described in Kessler to implement the x86 instruction set because the x86 instruction set has been widely accepted because of it's compatibility with a large amount of software as taught by Miller (Col. 2, lines 9-14). The Examiner added that Applicant's specification shows that integer instructions are inherent to the x86 instruction set (Page 27) and, therefore, when implementing the x86 instruction set in the coprocessor of Kessler, as previously described, the execution units would effectively operate as a "integer unit" as claimed.

Applicant respectfully disagrees and directs the Examiner's attention to arguments provided above in traversal of the rejections of claims 1 and 21. More specifically, claim 16, as amended herein, recites, an x86-compatible microprocessor that, among other features and limitations, has a cryptography unit, configured to execute one of the cryptographic operations responsive to receipt of a single, atomic cryptographic instruction that prescribes said one of the cryptographic operations, wherein said single, atomic cryptographic instruction is one of the instructions in an application program that are fetched from memory by fetch logic in said x86-compatible microprocessor, and wherein said x86-compatible microprocessor executes said application program. As has been argued above, the combination of Kessler, Colavin, and Miller teach away from utilization of a general purpose x86-compatible microprocessor to perform cryptographic operations. In all practical senses of combinations, the skilled artisan is led to utilize a coprocessor to perform these functions in order to offload the host processor. Furthermore, since none of the references teach or suggest a single, atomic cryptographic instruction, it does not follow that a combination of these references would make such a limitation obvious to one skilled in the art.

Since the noted elements and limitations are not taught, contemplated, or suggested by the combination of Kessler, Colavin, and Miller, it is requested that the rejection of claim 16 be withdrawn.

With respect to claims 17-20, these claims depend from claim 16 and add further limitations that are neither anticipated nor made obvious by the combination of Kessler, Colavin, and Miller. Accordingly, Applicant respectfully requests that the Examiner withdraw the rejections of claims 16-20.

CONCLUSIONS

Applicant believes this to be a complete response to all of the issues raised in the instant office action and further submits, in view of the amendments and arguments advanced above, that claims 1-25 are in condition for allowance. Reconsideration of the rejections is requested, and allowance of the claims is solicited.

Applicant also notes that any amendments made by way of this response, and the observations contained herein, are made solely for the purpose of expediting the patent application process in a manner consistent with the PTO's Patent business Goals (PBG), 65 Fed. Reg. 54603 (September 8, 2000), and are furthermore made without prejudice to Applicant under this or any other jurisdictions. It is moreover asserted that insofar as any subject matter might otherwise be regarded as having been abandoned or effectively disclaimed by virtue of amendments made herein and/or incorporated in attachments submitted with this response, Applicants wishes to reserve the right and hereby provides notice of intent to restore such subject matter and/or file a continuation application in respect thereof.

Applicant earnestly requests that the Examiner contact the undersigned practitioner by telephone if the Examiner has any questions or suggestions concerning this amendment, the application, or allowance of any claims thereof.

Respectfully submitted,
HUFFMAN PATENT GROUP, LLC

By: /Richard K. Huffman/

RICHARD K. HUFFMAN, P.E.
Registration No. 41,082
Tel: (719) 575-9998

Date: 04/27/2009